



## **Math 221: Numerical Analysis**

### **Chapter 2 - 2.1, 2.2, 2.3**

**Course Instructor:** Nahid Fatima

**Section #:** 1429

### **Project Contributors:**

SHOUG ALOMRAN 223410392

SHAHAD ABUNAYAN 223410189

MANAR ALTUWAIM 220410529

## Table Of Contents

<b>Abstract.....</b>	<b>3</b>
<b>Introduction.....</b>	<b>3</b>
<b>Solving the Problem (3 Methods).....</b>	<b>4</b>
Bisection.....	5
Newton–Raphson.....	6
Secant.....	7
<b>Comparison with MATLAB/Octave.....</b>	<b>8</b>
Figure 1. Numerical Output from Octave.....	8
<b>Applications in Computer Science and Engineering.....</b>	<b>9</b>
<b>Conclusion.....</b>	<b>10</b>
<b>References.....</b>	<b>10</b>

## Abstract

The present project will study three numerical techniques, namely Bisection, Newton–Raphson, and Secant Methods, for the solution of nonlinear equations. These three methods have become indispensable tools in computational mathematics since many nonlinear equations do not lend themselves to analytical solutions due to complexity or/and impossibility. Their convergence rate, accuracy, and computational efficiency are compared via MATLAB implementation. The Bisection method ensures convergence but with very slow performance, whereas the Newton–Raphson method converges rapidly when derivatives are known and an initial guess is close to the root. The Secant method offers a compromise of both, needing no derivative but depending on two initial guesses. The results presented demonstrate a balance between the reliability and speed of the numerical root-finding technique, which is the basis for solving problems in computer science and engineering applications, including optimization, data modeling, and simulation.

## Introduction

In many real-world problems in the fields of engineering and computer science, equations arise that cannot be solved analytically. Be it the simulation of physical systems, the optimization of neural networks, or simply the modeling of dynamic processes, there are usually demands for numerical methods that approximate unknown quantities.

This project focuses on solving nonlinear equations using three classical iterative methods: Bisection, Newton-Raphson, and Secant. Each of the three methods uses a different way of converging on a root - that is, the  $x$  value where  $f(x) = 0$ .

- Bisection method is a bracketing technique based on the Intermediate Value Theorem. It repeatedly halves an interval  $[a, b]$  where  $f(a)$  and  $f(b)$  have opposite signs until a desired accuracy is reached.



- Newton-Raphson Method. This is a calculus-based technique that utilizes derivatives so that the process of convergence may be accelerated. Given an initial guess  $x_0$ , the next approximation is given by:

$$x_{n+1} = x_n - (f(x)/f'(x))$$

### Solving the Problem (3 Methods)



### Bisection

$$f(x) = x^2 - 4$$

$$\boxed{x = 2}, \quad x = -2$$

1) Bisection Method  $[0, 3]$

$$a_0 = 0$$

$$f(0) = 0^2 - 4 = \boxed{-4} \quad (-ve)$$

$$b_0 = 3$$

$$f(3) = 9 - 4 = \boxed{5} \quad (+ve)$$

$$1) \quad \frac{0+3}{2} = \boxed{1.5}$$

$$f(1.5) = 1.5^2 - 4 = 2.25 - 4 = \boxed{-1.75} \quad (-ve)$$

$[1.5, 3]$

$$2) \quad \frac{1.5+3}{2} = 2.25$$

$$f(2.25) = 2.25^2 - 4 = 5.0625 - 4 = \boxed{1.0625} \quad (+ve)$$

$[1.5, 2.25]$

$$3) \quad \frac{1.5 + 2.25}{2} = \boxed{1.875}$$

$$f(1.875) = 1.875^2 - 4 = \boxed{-0.984375} \quad (-ve)$$

$[1.875, 2.25]$

4)

$$\frac{1.875 + 2.25}{2} = \boxed{2.0625}$$

$$f(2.0625) = 2.0625^2 - 4 = \boxed{0.25390625} \quad (+ve)$$

**Newton-Raphson**

2) Newton-Raphson Method

$$x_0 = 3$$

1)

$$f(3) = 9 - 4 = 5, \quad f'(3) = 6$$

$$x_1 = 3 - \frac{5}{6} = \sqrt{2.166667}$$

2)

$$f(2.166667) = 2.166667^2 - 4 = 0.694444$$

$$f'(2.166667) = 2 \times 2.166667 - 4 = 3.33333$$

$$x_2 = 2.166667 - \frac{0.694444}{3.33333} = \sqrt{2.0064103}$$

3)

$$f(2.0064103) = 0.0256821, \quad f'(2.0064103) = 4.0128205$$

$$x_3 = 2.0064103 - \frac{0.0256821}{4.0128205} = \sqrt{2.0000102}$$

4)  $f(2.0000102) = 0.00004096, \quad f'(2.0000102) = 4.0000205$

$$x_4 = 2.0000102 - \frac{0.00004096}{4.0000205} = \boxed{2}$$

Secant

3) Secant Method

$$x_{n+1} = x_n - \frac{f(x_n)}{f(x_n) - f(x_{n-1})} \cdot \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

$$x_0 = 0 \rightarrow f(0) = -4$$

$$x_1 = 3 \rightarrow f(3) = 5$$

$$1) x_2 = 3 - \frac{3-0}{5-(-4)} = 1.3333333$$

$$2) x_0 = 3, x_1 = 1.3333333$$

$$f(3) = 5, f(1.3333333) = -2.2222222$$

$$x_3 = 1.3333333 - (-2.2222222) \frac{1.3333333 - 3}{-2.2222222 - 5} = 1.8461538$$

3)

$$x_4 = 1.8461538 - (-0.5917160) \frac{1.8461538 - 1.3333333}{-0.5917160 - -2.2222222} = 2.0322581$$

$$4) f(1.8461538) = -0.5917160, f(2.0322581) = 0.1300728$$

$$x_5 = 2.0322581 - 0.1300728 \frac{2.0322581 - 1.8461538}{0.1300728 - (-0.5917160)} = 1.9987204$$

### Comparison with MATLAB/Octave

All three methods were implemented and tested using Octave, a MATLAB-compatible environment. The numerical results from all methods converged to the same root  $x = 2.000000$  for  $f(x) = x^2 - 4 = 0$ . The consistency between methods and the plotted curve verifies the correctness of the implementation and highlights the precision of MATLAB's numerical computation capabilities.

**Figure 1. Numerical Output from Octave**

```

--- Bisection Method ---
Iter 1: p = 2.000000, f(p) = 0.000000
Approximate root (Bisection) = 2.000000

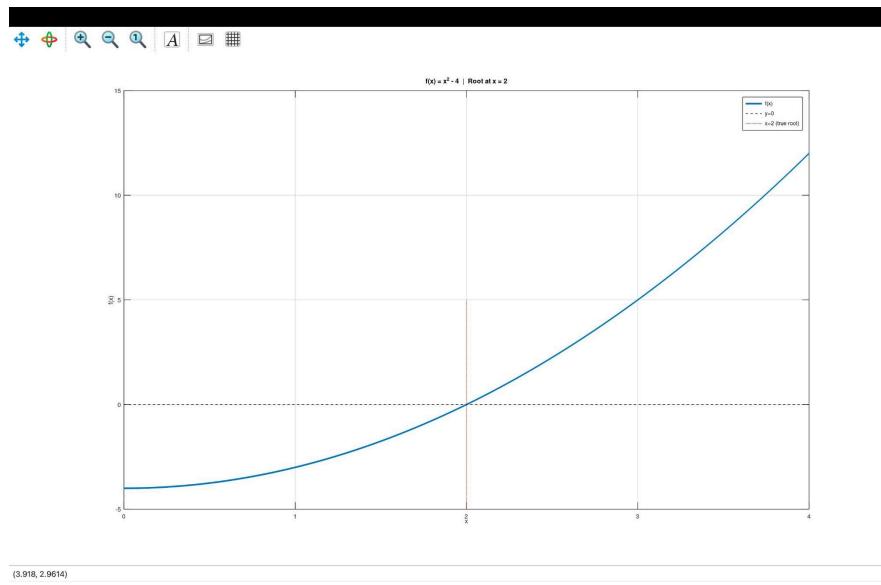
--- Newton-Raphson Method ---
Iter 1: x = 2.166667, f(x) = 0.694444
Iter 2: x = 2.006410, f(x) = 0.025682
Iter 3: x = 2.000010, f(x) = 0.000041
Iter 4: x = 2.000000, f(x) = 0.000000
Iter 5: x = 2.000000, f(x) = 0.000000
Approximate root (Newton-Raphson) = 2.000000

--- Secant Method ---
Iter 1: x = 1.750000, f(x) = -0.937500
Iter 2: x = 1.947368, f(x) = -0.207756
Iter 3: x = 2.003559, f(x) = 0.014248
Iter 4: x = 1.999953, f(x) = -0.000190
Iter 5: x = 2.000000, f(x) = -0.000000
Iter 6: x = 2.000000, f(x) = 0.000000
Approximate root (Secant) = 2.000000

--- Summary of Results ---
Bisection: 2.000000
Newton-Raphson: 2.000000
Secant: 2.000000

```

**Figure 2. Graph of  $f(x) = x^2 - 4$**



## Applications in Computer Science and Engineering

The significance of numerical root-finding extends much beyond pure mathematics:

1. Machine Learning and Optimization

Practical optimization algorithms, including gradient descent, are based on principles similar to Newton–Raphson for minimizing loss functions.

2. Computer Graphics and Simulation

Root-finding methods have many applications, including computing intersections when ray tracing, solving implicit curves, and approximating the equations of motion in physics engines.

3. Control Systems and Signal Processing

Engineers tune system parameters, design filters, and estimate the roots of transfer functions by applying iterative numerical algorithms.

4. Data Modeling and Computational Algorithms

Many algorithms, such as logistic regression, non-linear regression, polynomial fitting, require the solution of equations numerically when no closed-form solution exists.

## 5. Software Engineering and Algorithm Design

Numerical methods are part of algorithmic analysis for performance tuning, numerical stability, and error minimization.

In all, numerical methods provide an algorithmic backbone for computational problem-solving, aiding engineers and scientists in translating mathematical models into executable computer code that can be used to predict, simulate, and optimize real-world systems.

## Conclusion

This project illustrates how numerical root-finding methods bridge mathematical theory and computational practice. The Bisection method offers guaranteed convergence but at a very slow computational time. The Newton–Raphson method rapidly converges, using derivative information and a good initial guess, while the Secant method often allows for an effective middle ground, with fast results without derivatives and with sometimes sacrificed stability. Understanding these various trade-offs will prepare the Computer Science and Engineering student in choosing and implementing the right algorithm, based on problem constraints, making both accurate and efficient the computational solution at hand.

## References

1. Burden, R. L., & Faires, J. D. (2011). *Numerical Analysis* (9th ed.). Brooks/Cole.
2. Chapra, S. C., & Canale, R. P. (2015). *Numerical Methods for Engineers* (7th ed.). McGraw-Hill Education.
3. Lecture Slides: *Math 221 – Chapter 2: Bisection, Fixed Point, Newton–Raphson, and Secant Methods*, Prince Sultan University (2025).
4. MATLAB Documentation: *Root Finding – fzero Function* (MathWorks, 2025).
5. **Alomran, Shoug.** *Math221\_Numerical-Analysis* [Source Code]. GitHub, 2025. Available at: [https://github.com/Shoug-Alomran/Math221\\_Numerical-Analysis](https://github.com/Shoug-Alomran/Math221_Numerical-Analysis)